Lab 4 – Strings

PHP Strings: http://php.net/manual/en/book.strings.php

Aims:

- To be able to use various string functions and practice using control structures.

Getting Started:

Create a new folder named "**lab4**" under the "C:\htdocs" folder on your computer. Save today's work in this folder.

Task 1: Understanding string functions

Step 1:

Create a file "**strprocess.php**" that will receive an input from "**strform.php**" (in Step 2) via the POST method, remove all the vowels then output the resulting string. It should check if the input contains only the letters and spaces using regular expression, otherwise, it should generate an appropriate error message.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="description" content="Web application development" />
  <meta name="keywords" content="PHP" />
  <meta name="author"   content="Your Name" />
  <title>TITLE</title>
</head>
<body>
    <h1>Web Programming - Lab 4</h1>
    <?php // read the comments for hints on how to answer each item
      if (isset ($_POST["    (1)    "] &&     (1)     )){ // check if
      form data exists and is not empty
        $str = $_POST["    (2)    "];      // obtain the form data
        $pattern = "/^[A-Za-z ]+$/";      // set regular expression pattern
        if (          (3)          ) {       // check if $str with regular expression
          $ans = "";                        // initialise variable for the answer
          $len =        (4)         ;       // obtain length of string $str
          for ($i = 0; $i < $len; $i++) {       // checks all characters in $str
            $letter = substr ($str,  (5) , 1);    // extract 1 char using substr
            // check whether the extracted letter contains a number. Use strpos,
            is_numeric functions
            // if a position is not found, continue with
            the next statement
            if ((strpos ("AEIOUaeiou",  (6) )) === false){
                $ans = $ans . $letter; // concatenate letter to answer
            }
          }
          // generate answer after all letters are checked
          echo "<p>The word with no vowels is ", $ans, ".</p>";
        } else {                        // string contains invalid characters
        echo "<p>Please enter a string containing only letters or space.</p>";
        }
      } else {                          // no input
        echo "<p>Please enter string from the input form.</p>";
      }
    ?>
</body>
</html>
```

Step 2:
Create a file "**strform.php**" that contains a form with a single text box that allows a user to enter data, and submit it to "**strprocess.php**".

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="description" content="Web application development" />
  <meta name="keywords" content="PHP" />
  <meta name="author"   content="Your Name" />
  <title>TITLE</title>
</head>
<body>
    <h1>Web Programming Form - Lab 4</h1>
    <form action =       (7)       method =      (8)       >
        (9-10)
    </form>
</body>
</html>
```

Ensure that proper indentation is applied. Test in the browser, and validate your Web page, using a HTML validator: http://validator.w3.org.

Task 2: Practicing string functions

Background:
A perfect palindrome is a word or phrase that is identical forward or backward, such as the word "racecar". A standard palindrome is similar to a perfect palindrome except that spaces and punctuation are ignored. For example, "Madam, I'm Adam" is a standard palindrome because the characters are identical forward or backward, provided you remove the spaces and punctuation marks.

Step 1:
Create a file "**perfectpalindromeform.php**" that contains a form with a single text box that allows a user to enter a string, and submit it to "**perfectpalindrome.php**".

Step 2:
Create a file "**perfectpalindrome.php**" with a script that tests whether a word or phrase, entered by a user through the form is a **perfect** palindrome.

Hint: Use the `strrev()` function to reverse the input word or phrase and then use the `strcmp()` function to compare the original word or phrase with the reversed one.
Suggest also converting the strings to lower case, or upper case, before comparing them.

Take Note: You are to apply the following font type: Arial Narrow,Arial,sans-serif for all the output

Test in the browser, and check that both webpages are valid


Lab 04 Task 02 - Perfect Palindrome
String:
Check for Perfect Palindrome


Lab 04 Task 02 - Perfect Palindrome
The text you entered: 'raceCar' is a perfect palindrome!


Lab 04 Task 02 - Perfect Palindrome
Oopsie! The text you entered: 'raceCare' is NOT a perfect palindrome!

Task 3: Practicing the use of `str_replace()`

Step 1:
Save copies of the scripts created in Task 2 as "**standardpalindromeform.php**" and "**standardpalindrome.php**".

Step 2:
Modify the script to check for standard palindromes. For standard palindrome, first remove all the punctuation from the phrase using the `str_replace()` function before reversing the word or phrase and comparing it with the original one.

View in the browser, and check that both the webpages are valid.

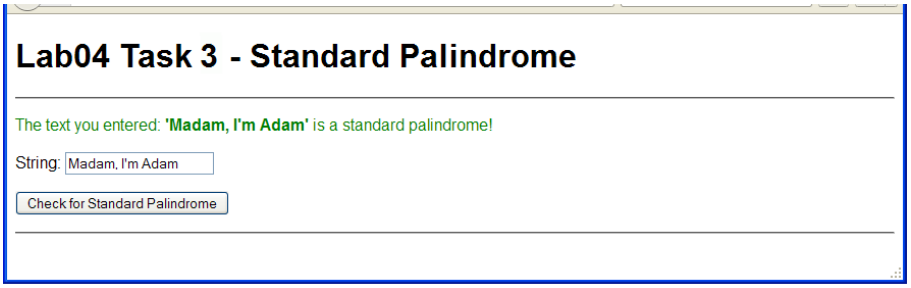
You should also try it out with the following palindromes:

- Eva, can I see bees in a cave?
- Was it a “cat” I saw?

Task 4: Extra Challenge

Combine the form "**standardpalindromeform.php**" and processing script "**standardpalindrome.php**", incorporating the form into a script "**standardpalindromeself.php**".
In other words, use a single webpage with the script that displays and also processes the form, i.e. the webpage calls itself.

Test in the browser, and check that it is valid.



Note:
If you want to prevent any problems that might be caused by users including html markup, such as < or >, as input in forms, then either replace them, or use the functions `htmlspecialchars()` or `htmlentities()` to convert these characters.

See http://php.net/manual/en/function.htmlspecialchars.php
and also see http://www.php.net/manual/en/function.htmlentities.php